



PATENT ABSTRACTS OF JAPAN

(11) Publication number: **09251379 A**(43) Date of publication of application: **22 . 09 . 97**

(51) Int. Cl. **G06F 9/06**
G06F 3/14
G06F 9/46

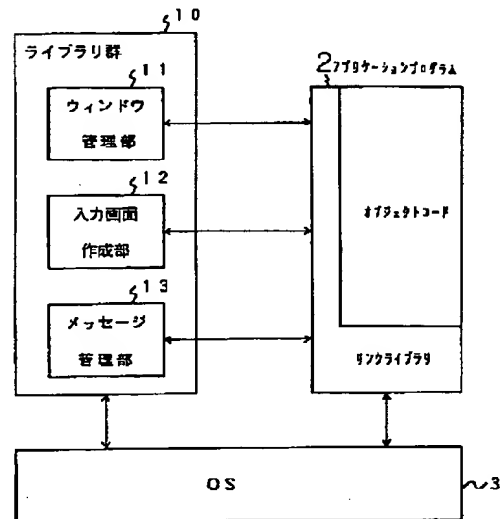
(21) Application number: **08061464**(22) Date of filing: **18 . 03 . 96**(71) Applicant: **TOSHIBA CORP TOSHIBA
SOFTWARE ENG KK**(72) Inventor: **SHIMIZU NOBUO
KONDO YUJI**(54) **COMPUTER SYSTEM**

(57) Abstract:

PROBLEM TO BE SOLVED: To easily transplant an application program not coping with a window system to the window system.

SOLUTION: The application program 2 not coping with the window system is linked with a library group 10 for transplanting and a window management part 11 prepares a window when the application program 2 is activated. Also, when the application program 2 requests the display of an input screen, an input screen preparation part 12 plots the input screen inside the window and a message management part 13 activates a message loop for receiving a message from an OS 3. Then, when data input from a user is present, by delivering the input data to the application program 2 and ending the message loop, control is returned to the application program 2.

COPYRIGHT: (C)1997,JPO



(19) 日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11) 特許出願公開番号

特開平9-251379

(43) 公開日 平成9年(1997)9月22日

(51) Int.Cl. ⁶	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 9/06	5 4 0		G 0 6 F 9/06	5 4 0 D
3/14	3 1 0		3/14	3 1 0 E
9/46	3 4 0		9/46	3 4 0 B

審査請求 未請求 請求項の数2 O L (全 12 頁)

(21) 出願番号 特願平8-61464

(22) 出願日 平成8年(1996)3月18日

(71) 出願人 000003078

株式会社東芝

神奈川県川崎市幸区堀川町72番地

(71) 出願人 000221133

東芝ソフトウェアエンジニアリング株式会
社

東京都青梅市新町1385番地

(72) 発明者 清水 伸夫

東京都青梅市末広町2丁目9番地 株式会
社東芝青梅工場内

(72) 発明者 近藤 祐史

東京都青梅市新町1385番地 東芝ソフトウ
ェアエンジニアリング株式会社内

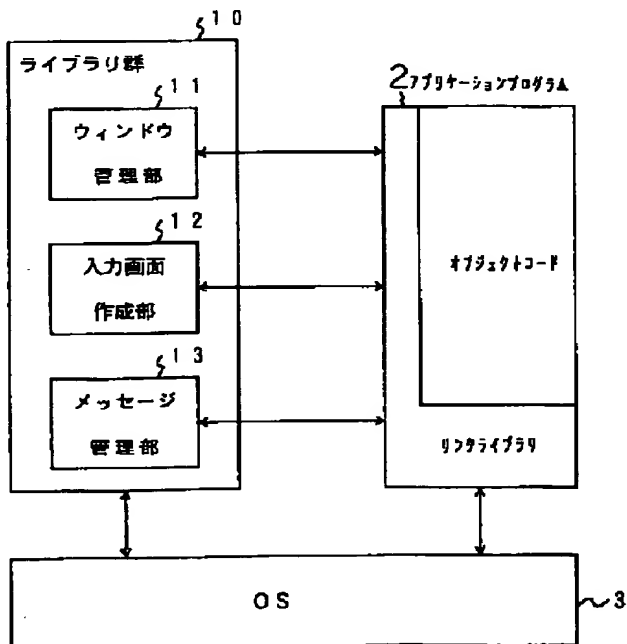
(74) 代理人 弁理士 鈴江 武彦

(54) 【発明の名称】 コンピュータシステム

(57) 【要約】

【課題】 ウィンドウシステムに対応していないアプリケーションプログラムをウィンドウシステムに容易に移植可能とするコンピュータシステムを提供する。

【解決手段】 ウィンドウシステムに未対応のアプリケーションプログラム2を移植用のライブラリ群10とリンクづけし、アプリケーションプログラム2の起動時に、ウィンドウ管理部11がウィンドウを作成する。また、アプリケーションプログラム2が入力画面の表示を要求したときには、入力画面作成部12がその入力画面をそのウィンドウ内に描画するとともに、OS3からのメッセージを受信するためのメッセージループをメッセージ管理部13が起動する。そして、ユーザからのデータ入力があったときに、その入力データをアプリケーションプログラム2に引き渡すとともに、メッセージループを終了させることによって制御をアプリケーションプログラム2に返却する。



【特許請求の範囲】

【請求項1】 同一画面上に少なくとも一つ以上のウインドウを表示し、それぞれのウインドウで別々のアプリケーションプログラムを実行可能なオペレーティングシステムであって、前記ウインドウ内で動作中のアプリケーションプログラムに各種メッセージを送信する機能を有するオペレーティングシステムを適用してなるコンピュータシステムにおいて、

前記メッセージの受信機能を含む前記ウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムと関係づけられ、

前記非対応アプリケーションプログラムの起動時に、前記オペレーティングシステムにウインドウの作成を要求するウインドウ作成手段と、

前記非対応アプリケーションプログラムが入力画面の表示を要求したときに、その入力画面を前記ウインドウ作成手段により作成されたウインドウ内に描画する入力画面作成手段と、

前記入力画面作成手段による前記ウインドウ内への前記入力画面の描画と同期して、前記オペレーティングシステムからのメッセージを受信するためのメッセージループを起動するメッセージループ起動手段と、

前記メッセージループ起動手段により受信したメッセージが、前記非対応アプリケーションプログラムが表示要求した入力画面に対応する入力データを含むものであったときに、その入力データを前記非対応アプリケーションプログラムに送信するとともに前記メッセージループを終了させるメッセージ処理手段と、

前記非対応アプリケーションプログラムの終了時に、前記オペレーティングシステムにウインドウの解放を要求するウインドウ解放手段とを有してなるライブラリ群を具備し、

前記ウインドウ対応がなされていない非対応アプリケーションプログラムを前記ライブラリ群と関係づけるのみで前記ウインドウ環境下に移植可能とすることを特徴とするコンピュータシステム。

【請求項2】 前記メッセージ処理手段は、前記受信したメッセージが前記ウインドウの移動やサイズ変更を含む前記ウインドウ制御のためのメッセージであったときに、前記非対応アプリケーションプログラムに代替してその処理を実行することを特徴とする請求項1記載のコンピュータシステム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】この発明は、ディスプレイを複数のウインドウで区切り、それぞれのウインドウで別々のアプリケーションプログラムを実行するオペレーティングシステムを適用してなるコンピュータシステムに係り、特にこのウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムをウイ

ンドウシステムに容易に移植することを可能とするコンピュータシステムに関する。

【0002】

【従来の技術】近年、デスクトップをはじめとして、ラップトップ、ノートブックおよびパームトップなど、種々のパーソナルコンピュータが開発されており、その普及は目覚ましいものがある。そして、これらの普及に伴って、コンピュータシステムとユーザとのインタフェースに様々な工夫がなされてきている。

10 【0003】従来では、CUI (Character User Interface) などと称されるインタフェースが主流であり、システムで実行中のアプリケーションプログラムがユーザにデータの入力を促す入力画面を表示し、一方で、ユーザがキーボードを介してコマンドを入力することによってインタフェースを確保するといったものであった。

20 【0004】しかしながら、このような方法によると、たとえば、ユーザ自身がアプリケーションプログラムと対話するためのコマンドを覚えておかなければならないなどといった問題があり、あまり使い勝手がよいとはいえなかった。

30 【0005】そして、このような問題を解決するものとして、GUI (Graphical User Interface) などと称されるインタフェースが登場してきた。このGUIは、画面上に絵やアイコンなどを表示して、直感的に分かりやすく操作できるようにしたものであり、マウスなどのポインティングデバイスによって、表示されたアイコンを指定することによって操作を進めていくことが可能である、そして、最近では、このGUIが、コンピュータシステムとユーザとのインタフェースの主流となってきている。

40 【0006】ウインドウシステムとは、ディスプレイの画面をいくつかの窓 (ウインドウ) に分け、それぞれのウインドウで別々のアプリケーションを動作させることのできるシステムをいう。このウインドウには、画面を上下左右に分割し、その一つ一つをウインドウとして各アプリケーションプログラムに割り当てるものや、ウインドウの重なりを許可して複数のウインドウを作成し、これら複数のウインドウそれぞれに各アプリケーションプログラムを割り当てるなどといったものが存在する。

50 【0007】一方、このウインドウシステムを実現するOS配下でアプリケーションプログラムを動作させるためには、アプリケーションプログラム自身もウインドウ対応がなされていなければならない。

【0008】ウインドウシステムでは、前述したようにGUIが採用されており、これによってユーザは、マウスなどのポインティングデバイスによってほとんどの操作を行なう。そして、このウインドウシステムでは、ウインドウの移動やサイズ変更などもポインティングデバイスで行なえるようになっており、OSは、入力データ

をはじめ、これらの情報をも、そのウインドウ内で動作中のアプリケーションプログラムにメッセージとして通知する仕組みとなっている。

【0009】すなわち、このような環境下で動作するアプリケーションプログラムは、OSからのメッセージを取得するメッセージ受信機能を備え、かつこのメッセージに対応したメッセージ処理機能を備えていなければならない。

【0010】しかしながら、ウインドウシステムに対応していないアプリケーションプログラムを資源として活用するユーザは依然として多く、また、このようなアプリケーションプログラムを前述したようなウインドウシステムの環境下に移植するためには、ソースプログラムレベルでの修正が必要であり、オブジェクト提供のアプリケーションプログラムの場合、エンドユーザでは移植が不可能であるといった問題があった。

【0011】

【発明が解決しようとする課題】このように、最近のパーソナルコンピュータに適用されるOSは、ウインドウシステムを前提としているものがほとんどであるが、このようなウインドウシステムに対応していないアプリケーションプログラムを資源として活用するユーザは依然として多く、また、このようなアプリケーションプログラムを前述したようなウインドウシステムの環境下に移植するためには、ソースプログラムレベルでの修正が必要であり、オブジェクト提供のアプリケーションプログラムの場合、エンドユーザでは移植が不可能であるといった問題があった。

【0012】この発明はこのような実情に鑑みてなされたものであり、ウインドウシステムで動作するための対応がなされていない非対応アプリケーションプログラムを容易にウインドウシステムに移植することを可能とするコンピュータシステムを提供することを目的とする。

【0013】

【課題を解決するための手段】この発明は、同一画面上に少なくとも一つ以上のウインドウを表示し、それぞれのウインドウで別々のアプリケーションプログラムを実行可能なオペレーティングシステムであって、前記ウインドウ内で動作中のアプリケーションプログラムに各種メッセージを送信する機能を有するオペレーティングシステムを適用してなるコンピュータシステムにおいて、前記メッセージの受信機能を含む前記ウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムと関係づけられ、前記非対応アプリケーションプログラムの起動時に、前記オペレーティングシステムにウインドウの作成を要求するウインドウ作成手段と、前記非対応アプリケーションプログラムが入力画面の表示を要求したときに、その入力画面を前記ウインドウ作成手段により作成されたウインドウ内に描画する入力画面作成手段と、前記入力画面作成手段による前記

ウインドウ内への前記入力画面と同期して、前記オペレーティングシステムからのメッセージを受信するためのメッセージループを起動するメッセージループ起動手段と、前記メッセージループ起動手段により受信したメッセージが、前記非対応アプリケーションプログラムが表示要求した入力画面に対応する入力データを含むものであったときに、その入力データを前記非対応アプリケーションプログラムに送信するとともに前記メッセージループを終了させるメッセージ処理手段と、前記非対応アプリケーションプログラムの終了時に、前記オペレーティングシステムにウインドウの解放を要求するウインドウ解放手段とを有してなるライブラリ群を具備し、前記ウインドウ対応がなされていない非対応アプリケーションプログラムを前記ライブラリ群と関係づけるのみで前記ウインドウ環境下に移植可能とすることを特徴とする。

【0014】この発明によれば、ウインドウシステム環境下にウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムを移植する際、ユーザは、そのアプリケーションプログラムと本発明の移植用のライブラリ群とをリンクする。

【0015】たとえば、アプリケーションプログラムがDLL(Dynamic Link Library)を使用する場合、そのアプリケーションプログラムがどのDLLを使用するかを決定するために、アプリケーションプログラムの作成時にインポートライブラリをリンクする。そして、OSは、このインポートライブラリのリンクにより、そのアプリケーションプログラムがどのDLLを利用して動作するものなのかを認識することができる。したがって、この場合、非対応アプリケーションプログラムは、このインポートライブラリのリンクによって、移植用のライブラリ群(DLL)と関係づけられることになる。

【0016】従来のアプリケーションプログラムでは、画面に入力画面を表示して、この画面に対してユーザが入力した情報を入力データとして取り込み、この取り込んだ入力データに基づいて処理を進行させていくといった逐次的な動作が行なわれていく。

【0017】そこで、この発明では、移植用のライブラリ群が、以下のタイミングで以下のウインドウ処理を代替することによって、非対応アプリケーションプログラムをウインドウシステムに移植する。

(1) 起動時

OSへのウインドウの作成の要求。

(2) 入力画面の表示時

ウインドウ内での入力画面の描画。

【0018】OSからのメッセージを受信するためのメッセージループの起動

(3) OSからのメッセージ受信時。

a) 入力データの場合

アプリケーションプログラムへの入力データの転送。

【0019】メッセージループの終了（これにより、アプリケーションプログラムに制御が返却される）。

b) 移動などのウインドウ制御の場合
メッセージに応じたウインドウ処理。

(4) 終了時

OSへのウインドウの解放の要求。

【0020】これにより、OSからはウインドウシステム対応のアプリケーションプログラムとして認識され、かつアプリケーションプログラムは従前のインタフェースで入力データを取り込みながら処理を行なえることになり、さらにアプリケーションプログラムの修正などを一切必要としない。

【0021】

【発明の実施の形態】以下、図面を参照してこの発明の一実施形態を説明する。図1は、本実施形態のコンピュータシステムの概略構成を示すブロック図である。

【0022】図1に示すように、本実施形態のコンピュータシステムは、OS（オペレーティングシステム）3によって全体が制御されている。また、このOS3配下で動作するアプリケーションプログラム2は、DLL（Dynamic Link Library）1に格納された複数の機能オブジェクト（ライブラリ群10a～10b）のうち、必要とする機能オブジェクトと関連づけられて動作する。そして、本実施形態の特徴とする移植用のライブラリ群も、このDLL1に格納され、ウインドウシステム非対応のアプリケーションプログラムは、この移植用のライブラリ群と関連づけられることになる。

【0023】なお、これらアプリケーションプログラム2と、DLL1中の意図するライブラリ群との関係づけは、アプリケーションプログラム2の作成時に、インポートライブラリ（リンクライブラリ）をリンクすることによって実現される。そして、OS3は、このインポートライブラリのリンクにより、アプリケーションプログラム2それぞれが、どのライブラリ群能を利用して動作するものなのかを認識する。

【0024】図2には、本実施形態の移植用のライブラリ群の概略構成が示されている。図2に示すように、本実施形態の移植用ライブラリ群10は、ウインドウ管理部11、入力画面作成部12およびメッセージ管理部13を具備してなる。

【0025】ウインドウ管理部11は、非対応アプリケーションプログラム2の稼働時に、OS3に対してウインドウの作成を要求するとともに、非対応アプリケーションプログラム2の終了時に、OS3に対してウインドウの解放を要求する。

【0026】入力画面作成部12は、非対応アプリケーションプログラム2が入力画面の表示を要求したときに、ウインドウ管理部11によって作成されたウインド

ウ内にその入力画面を描画する。

【0027】そして、メッセージ管理部13は、非対応アプリケーションプログラム2による入力画面の表示要求に基づいて入力画面作成部12が入力画面を描画したときに、OS3からのメッセージを受信するためのメッセージループを起動し、たとえばウインドウの移動やサイズ変更などといったメッセージの種別に応じた処理を行なうとともに、入力データの非対応アプリケーションプログラム2への引き渡しおよびメッセージループの終了を行なう。

【0028】ここで、図3を参照してウインドウシステムへの対応がなされていないアプリケーションプログラムの従来の環境での動作手順を説明しておく。まず、非対応アプリケーションプログラムは、OSに対して入力画面の作成と表示を要求する（ステップF1）。ユーザがこの入力画面に対応したデータ入力を行なうと、そのデータはOSから非対応アプリケーションプログラムに引き渡され、非対応アプリケーションプログラムは、この引き渡された入力データに基づいた処理を実行する（ステップF2）。

【0029】すなわち、非対応アプリケーションプログラムは、ステップF1～ステップF2を必要なデータ分だけ繰り返す、逐次的な処理を進行させていく。図4には、ウインドウシステムへの対応がなされていないアプリケーションプログラムの従来の環境での動作原理が示されている。

【0030】非対応アプリケーションプログラムが、OSに対して入力画面の作成と表示を要求した後（図4の（1））、OSからユーザの入力したデータが送信されると（図4の（2））、非対応アプリケーションプログラムは、この送信された入力データに基づいた処理を実行する（図4の（3））。

【0031】同様に、次のデータが必要な場合には、OSに対して入力画面の作成と表示を要求し（図4の（4））、この要求に対して返送された入力データ（図4の（5））に基づいて処理を実行する。

【0032】このように、従来の非対応アプリケーションプログラムは、必要なデータを受け取りつつ、逐次的な処理を進行させていく。次に、図5を参照してウインドウシステムへの対応がなされたアプリケーションプログラムの動作手順を説明する。

【0033】まず、対応アプリケーションプログラムは、OSからのメッセージを入力するためのメッセージループを起動する（ステップG1）。このメッセージループには、メッセージの受信処理と、そのメッセージの種別に応じたディスパッチ処理とが含まれている。

【0034】このメッセージループ起動後、対応アプリケーションプログラムは、OSからのメッセージを待機し（ステップG2）、メッセージを受信したときに（ステップG3のY）、このメッセージが終了を指示するメ

10

20

30

40

50

ッセージかどうか判定し（ステップG4）、終了を指示するメッセージでなければ（ステップG4のN）、そのメッセージの種別に基づく処理を実行する（ステップG5）。

【0035】したがって、このとき、ウインドウの移動やサイズ変更などのウインドウ制御に関するメッセージを受信したときには、その処理を行なうモジュールをディスパッチし、ユーザなどからの入力データを含むメッセージを受信したときには、そのデータの処理を行なうモジュールをディスパッチする。

【0036】図6には、ウインドウシステムへの対応がなされたアプリケーションプログラムの動作原理が示されている。対応アプリケーションプログラムが、OSからのメッセージを入力するためのメッセージループを起動すると（図6の（1））、これ以降、ユーザからのデータ入力をはじめ、ウインドウの移動やサイズ変更など、イベントが発生するたびにOSからメッセージが送信され、その都度、そのメッセージに基づいた処理が行なわれる（図6の（2）～（3）、（4）～（5）、（6）～）。そして、この処理は、対応アプリケーションプログラムが終了するまで繰り返される。

【0037】ここで、本実施形態の環境（ウインドウシステム環境）におけるウインドウシステムへの対応がなされていないアプリケーションプログラムの動作手順を説明する。

【0038】本実施形態のコンピュータシステムでは、図2に示したように、非対応アプリケーションプログラム2を移植用のライブラリ群10とリンクさせることによって、実行プログラムを作成する。以下、この実行アプリケーションの動作手順を説明する。

【0039】図7乃至図11には移植用のライブラリ群10とリンクされた非対応アプリケーションプログラム2との動作手順が示されている。非対応アプリケーションプログラム2は、起動すると、まず、初期処理を実行する（図7のステップA1）。このとき、非対応アプリケーションプログラム2では、「Open」などといった、起動時に必須の関数を発行することになり、この関数の発行は、OS3の制御によってリンクづけられた移植用のライブラリ群10に通知されることになる。

【0040】この通知を受けたときの移植用のライブラリ群10の動作を図8に示す。この通知を受け取った移植用のライブラリ群10では、ウインドウ管理部11が、非対応アプリケーションプログラム2がウインドウシステム環境で動作するための各種資源を確保する（図8のステップB1）。そして、ウインドウ管理部11は、非対応アプリケーションプログラム2が動作するためのウインドウの作成を非対応アプリケーションプログラム2に変わって実行する（図8のステップB2）。

【0041】この後、非対応アプリケーションプログラム2が、データを入力するための画面表示を要求する関

数を発行すると（図7のステップA2）、同様にこの関数の発行は、OS3の制御によってリンクづけられた移植用のライブラリ群10に通知され、この通知を受け取った移植用のライブラリ群10では、入力画面作成部12が、図9に示すような動作を行なう。

【0042】すなわち、OS3に対してウインドウプロシージャの呼び出しを実行し（図9のステップC1）、非対応アプリケーションプログラム2から受け取った表示要求に基づいたウインドウプロシージャ内での描画処理を実行する（図9のステップC2）。

【0043】そして、この表示した画面を介したデータの inputs が非対応アプリケーションプログラム2から要求されると、移植用のライブラリ群10では、メッセージ管理部13が、図10に示す処理を実行する。

【0044】まず、メッセージ管理部13は、OS3がウインドウ管理部11によって作成されたウインドウに対して送信するメッセージを受信するためのメッセージループを起動する（図10のステップD1）。これにより、入力画面作成部12によって描画された表示画面に対応してユーザが入力したデータを含むメッセージを受信可能になるとともに、ウインドウの移動やサイズ変更などを指示するメッセージも受信可能となる。

【0045】そして、メッセージ管理部13は、受信されたメッセージに応じてディスパッチを実行し、その処理を行なう（図10のステップD2）。なお、ウインドウプロシージャでのキー入力の確認をした結果、ユーザからの入力データであった場合には、非対応アプリケーションプログラム2が参照可能な内部キーバッファに入力されたキーコードを登録し（図10のステップD

3）、メッセージループを終了させる（図10のステップD4）。すなわち、このメッセージループの終了によって、非対応アプリケーションプログラム2側に制御が返却され、非対応アプリケーションプログラム2側での実行が再開されることになる。

【0046】そして、非対応アプリケーションプログラム2が終了処理を実行したときに（図7のステップA4）、移植用のライブラリ群10では、ウインドウ管理部11が、図11に示すような手順で処理を行なう。

【0047】ウインドウ管理部11は、非対応アプリケーションプログラム2の終了時に確保した各種資源を解放し（図11のステップE1）、OS3に対してウインドウの破壊を要求する（図11のステップE2）。

【0048】これにより、非対応アプリケーションプログラム2は、ウインドウシステムの環境を意識せずに動作することができる結果、ソースレベルでの修正などが一切必要なくなり、かつOS3からはウインドウシステム対応のアプリケーションプログラムとして認識されることになる。

【0049】このときの動作原理を図12を参照して説明する。アプリケーションプログラム2が起動され、初

10

20

30

40

50

期化处理が実行されると(図12の(1))、移植用ライブラリ群10のウインドウ管理部11は、OS3に対してアプリケーションプログラム2用のウインドウの作成などを要求する(図12の(2))。ここでのウインドウ管理部11の処理終了がアプリケーションプログラム2に通知されると(図12の(3))、次に、アプリケーションプログラム2は、データ入力用画面作成要求を移植用ライブラリ群に通知する(図12の(4))。この通知を受けた移植用ライブラリ群10では、入力画面作成部12が、先に作成したウインドウのどの位置に文字や罫線を描画すべきかを判断してそのデータ入力用画面を作成されたウインドウ内に描画する(図12の(5))。このとき、入力画面作成部12は、文字を書く要求などを状況に応じて文字単位や文字列単位でOS3に発行する。そして、入力画面作成部12は、描画要求のすべてが満たされると、アプリケーションプログラム2に対して処理を返却する(図12の(6))。

【0050】アプリケーションプログラム2は、その処理中にデータの入力が必要になると、移植用ライブラリ群10に対してデータ入力要求を発行する(図12の(7))。この要求を受け付けた移植用ライブラリ群10では、メッセージ管理部13が、OS3からメッセージを受信するためのメッセージループを起動する(図12の(8))。

【0051】なお、このとき、ウインドウシステムでは、要求したデータ入力以外のメッセージ、たとえばウインドウの移動やサイズの変更などを示すメッセージを受信することも考えられる(図12の(9))。この場合、移植用ライブラリ群10では、この受信メッセージの要求を満たす処理を実行し(図12の(10))、さらに期待するデータの入力を待機する。

【0052】また、その期待するデータを含むメッセージが受信された場合(図12の(11))、メッセージ管理部13は、メッセージループを終了させるとともに(図12の(12))、このメッセージに含まれる入力データをアプリケーションプログラム2に引き渡す(図12の(13))。これにより、アプリケーションプログラム2に制御が返却される。

【0053】そして、アプリケーションプログラム2は、この受け取った入力データに基づく処理を実行する。これ以降、たとえばユーザからのデータ入力が必要とする場合には、再度同様の処理を繰り返す。そして、アプリケーションプログラム2が終了処理を実行したときに、移植用ライブラリ群10のウインドウ管理部11は、アプリケーションプログラム2の起動時に確保した各種資源の解放や作成したウインドウの破壊処理を実行する。

【0054】なお、本実施形態では、メッセージループを終了させることによって非対応アプリケーションプログラム2に制御を返却するため、たとえば制御を非対応

アプリケーションプログラム2に返却した後にウインドウの移動が指示された場合には、このメッセージの受信が即座には行なえないことになるが、次に非対応アプリケーションプログラム2からデータの入力要求が行なわれた際に、このウインドウ処理が真っ先に実行されることになるため、問題はないと思われる。

【0055】これにより、ユーザは、ウインドウシステムに対応していないアプリケーションプログラムを容易かつ安価にウインドウシステムに移植することができることになる。

【0056】

【発明の効果】以上詳述したように、この発明によれば、ウインドウ対応がなされていない非対応アプリケーションプログラムを移植用のライブラリ群と関係づけるのみでウインドウ環境下に移植することができるため、ソースレベルでの修正などが一切不要となり、かつウインドウシステムに対応させるための知識も必要とせず容易かつ安価に移植作業を行なうことができる。

【図面の簡単な説明】

20 【図1】この発明の実施形態のコンピュータシステムの概略構成を示すブロック図。

【図2】同実施形態の移植用のライブラリ群の概略構成を示す図。

【図3】ウインドウシステムへの対応がなされていないアプリケーションプログラムの従来環境での動作手順を説明するフローチャート。

【図4】ウインドウシステムへの対応がなされていないアプリケーションプログラムの従来環境での動作原理を示す図。

30 【図5】ウインドウシステムへの対応がなされたアプリケーションプログラムの動作手順を説明するフローチャート。

【図6】ウインドウシステムへの対応がなされたアプリケーションプログラムの動作原理を示す図。

【図7】同実施形態の環境でのウインドウシステムへの対応がなされていないアプリケーションプログラムの動作手順を説明するフローチャート。

【図8】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

40 【図9】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

【図10】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

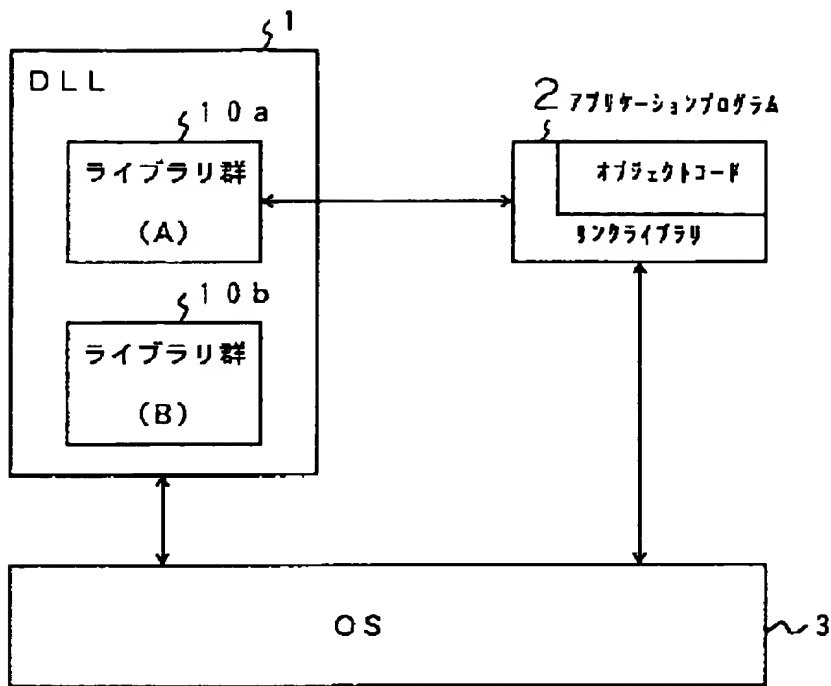
【図11】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

【図12】同実施形態の動作原理を説明する図。

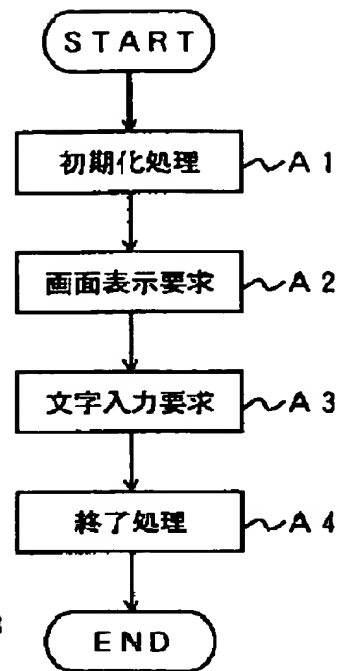
【符号の説明】

1…DLL、2…アプリケーションプログラム、3…OS、11…ウインドウ管理部、12…入力画面作成部、13…メッセージ管理部。

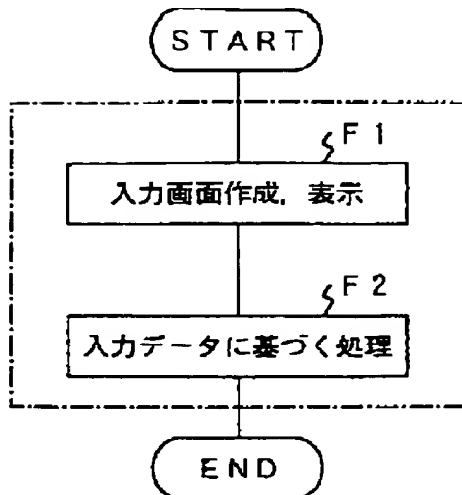
【図1】



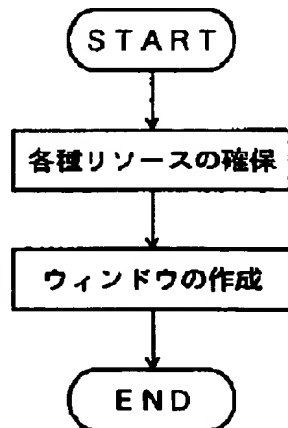
【図7】



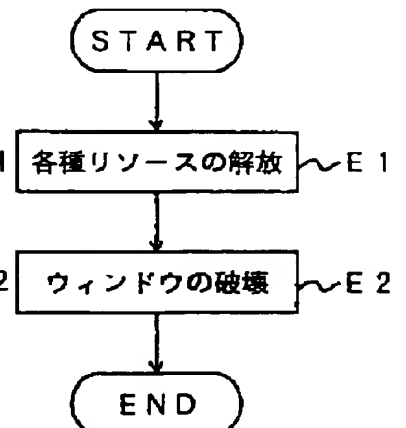
【図3】



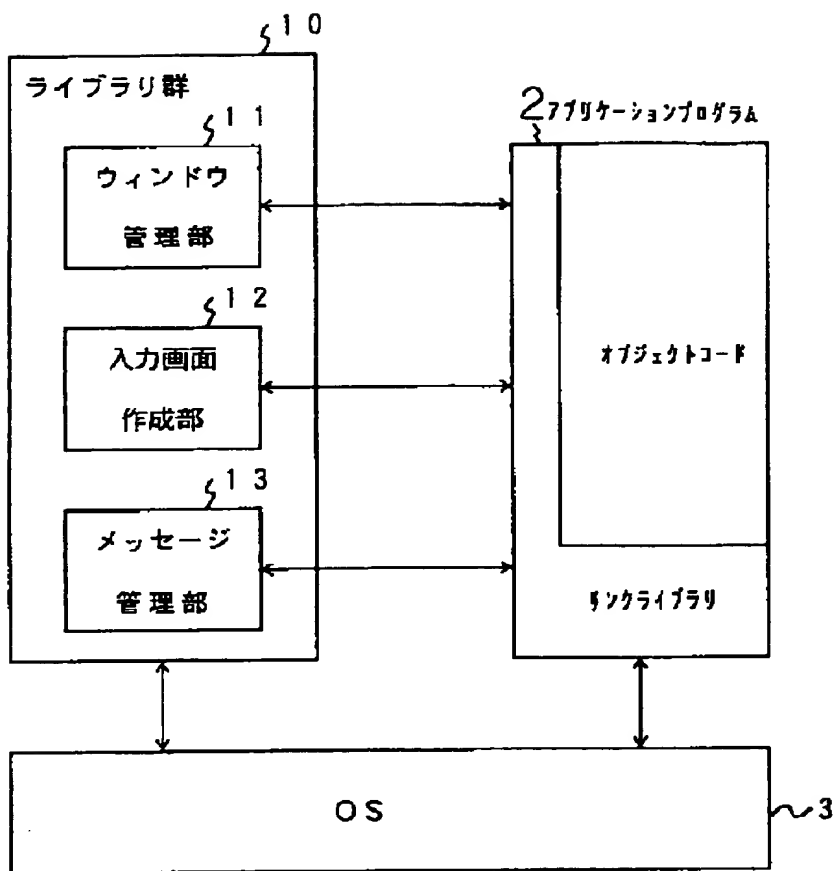
【図8】



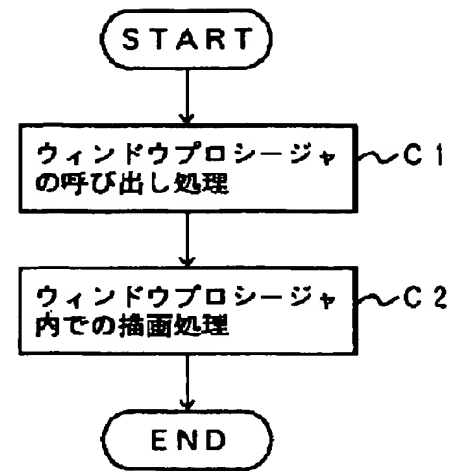
【図11】



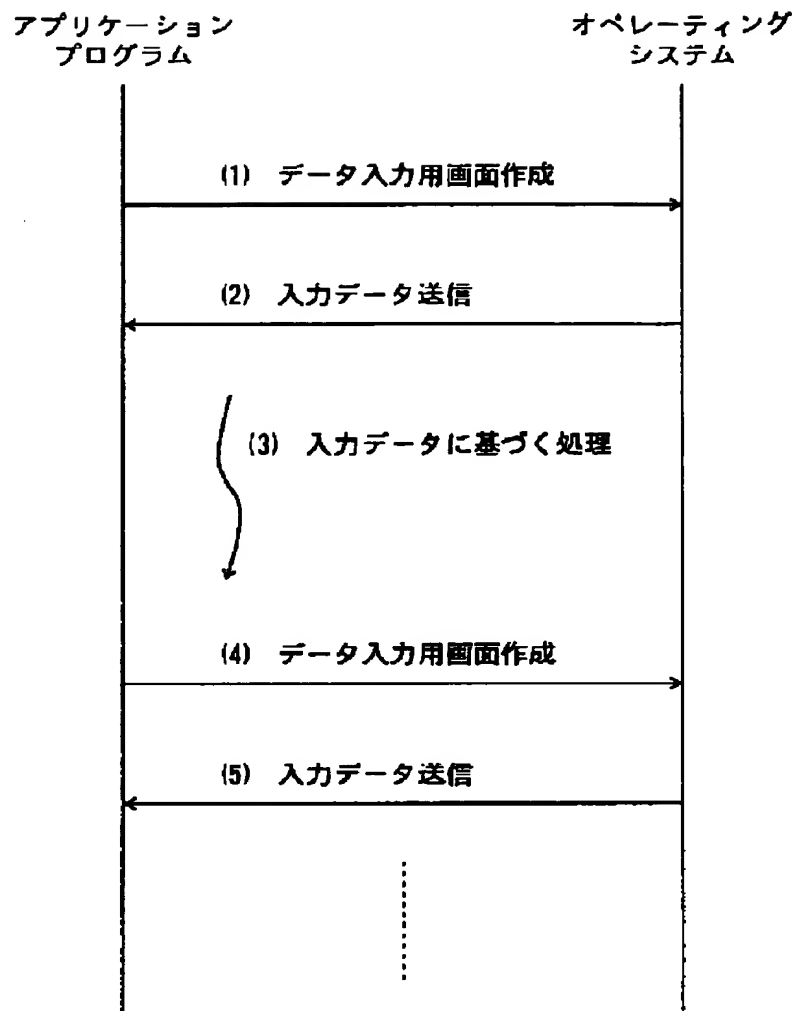
【図2】



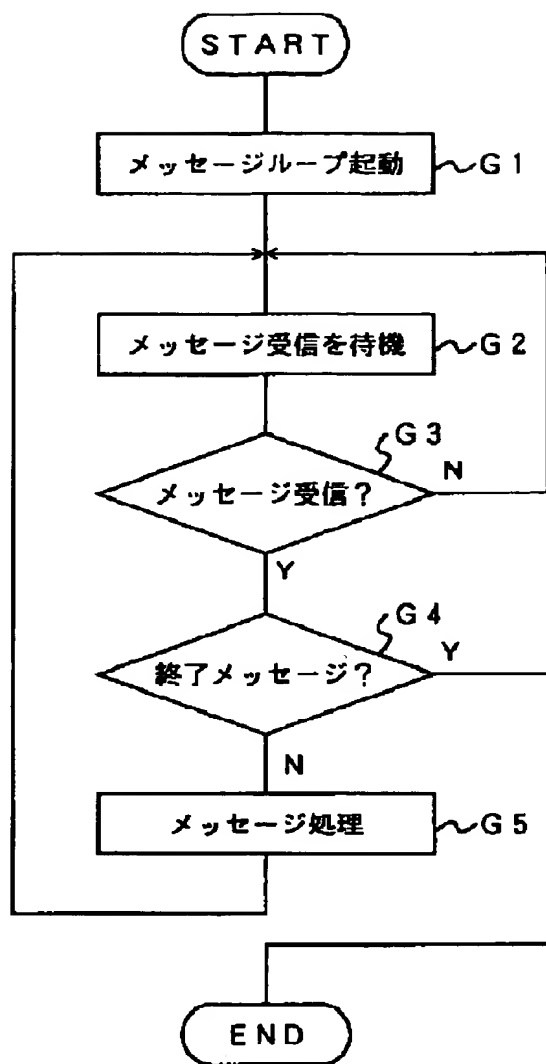
【図9】



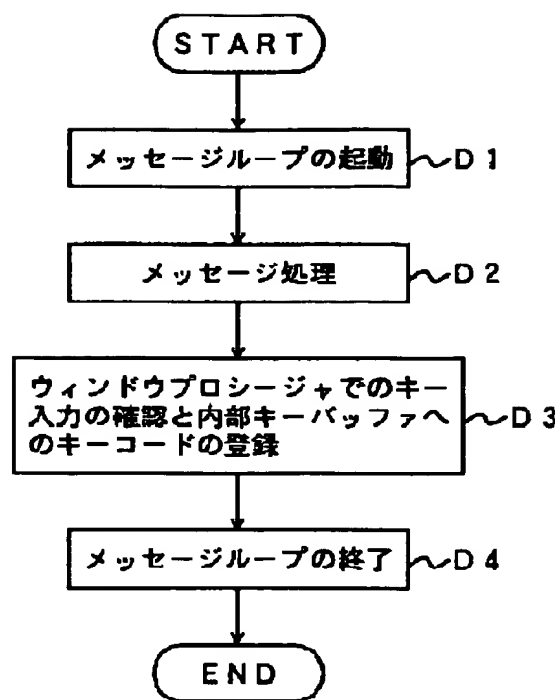
【図4】



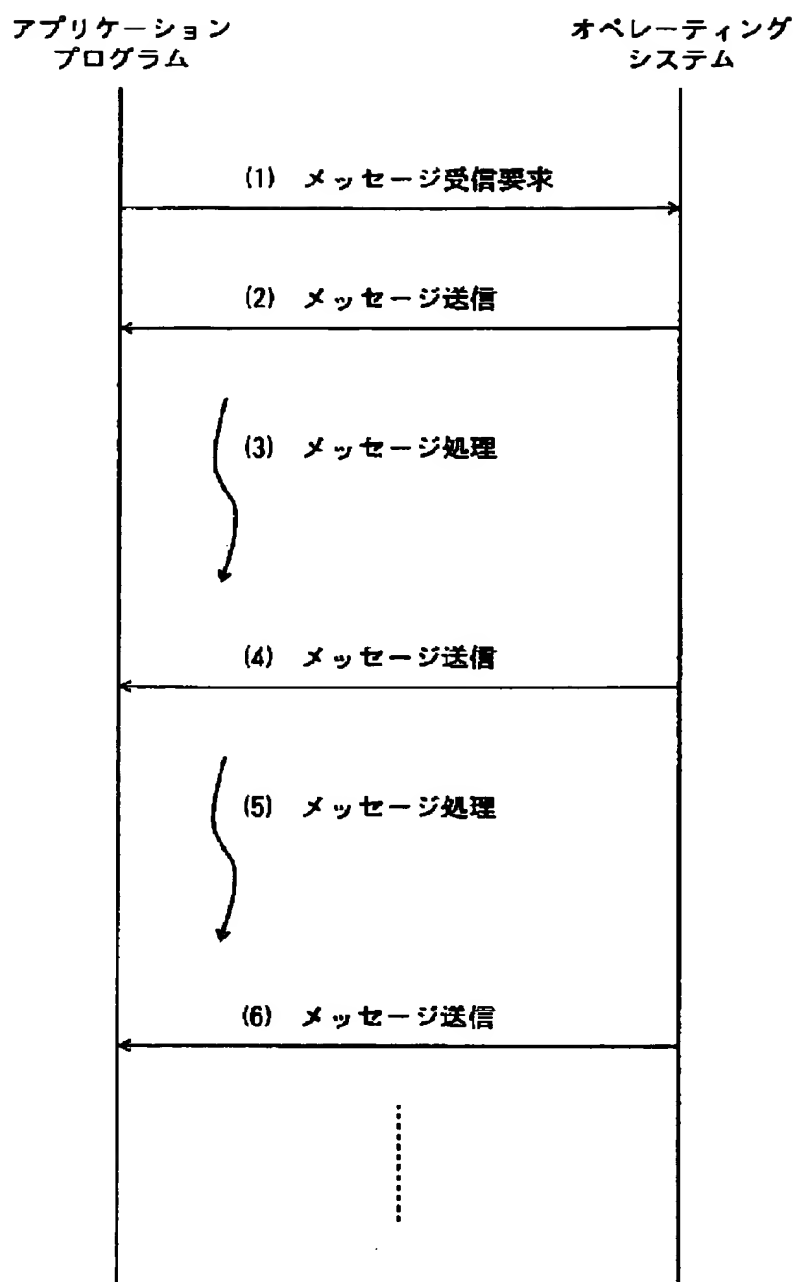
【図5】



【図10】



【図6】



【図12】

